


Article

Open Source Data Logging and Data Visualization for an Isolated PV System

Bojian Jiang *  and M. Tariq Iqbal *

Faculty of Engineering and Applied Science, Memorial University of Newfoundland (MUN),
St. John's, NL A1B 3X5, Canada

* Correspondence: bjiang@mun.ca (B.J.); tariq@mun.ca (M.T.I.)

Received: 13 March 2019; Accepted: 2 April 2019; Published: 11 April 2019



Abstract: Monitoring the operation of an isolated photovoltaic (PV) system needs both data loggers and web transfer to collect the sensor data. The data includes the measurement of the voltage and current of the PV system and for local weather. The PV system in Memorial University of Newfoundland (MUN) is 5 m away from the window, where the weather data is collected. In reality, PV systems are approximately 25–50 m away from the weather sensors. It is, therefore, more meaningful to realize the sensor communications by wireless transfer than long cables, which can significantly reduce the cables of a large PV system with long distances among sensors. The PC receives all the sensor data and transfers them to a web server (Thingspeak). A web server is applied to monitor the operation of the system instead of a local server when its users are far away from the location, even though the local server allows more frequent data logging (once per second). The data transformation between the PC and the web server must guarantee the stability and robustness of the program. The system alarm that reports the disconnection failure is also necessary to notify the users. This paper will first introduce the general system setup, then present each part of the system in detail, and finally, analyze the collected data.

Keywords: PV system; data logger; radio frequency; Arduino; web transfer; Thingspeak; MATLAB

1. Introduction

For an isolated photovoltaic (PV) system, it is necessary to monitor the system when its users are away from the location. Typically, such systems need data logging and SCADA systems to visualize the collected data for system monitor [1]. Rezk et al. [2] and Sánchez-Pacheco et al. [1] used a data-acquisition device and programmable logic controller (PLC), respectively, as the data loggers. However, PLC devices and data-acquisition devices are too expensive for remote residences. Many low-cost data loggers are designed due to this reason, such as the loggers designed by Munandar and Syamsi [3], Ramdan et al. [4], and Hadiatna et al. [5]. All these data loggers are designed based on microcontrollers or FPGAs, which are much cheaper than industrial PLC. Although economic data loggers are presently used for isolated systems, their circuits are very complicated. Thus, a data logging system with an easy circuit design is required.

In addition to circuit design, the sensors in references [1–5] mainly communicate through cables. Such data logging systems could quickly become disordered and difficult to maintain. Touati et al. [6] solved the problem by using XBee, which is a wireless mesh network. Although the mesh network is reliable for data transmission with a low energy cost, the transmission range of XBee is less than 100 m, which is much shorter than radio frequency transmission [7]. According to Terashimila et al. [8], radio frequency can achieve reliable communication within 1 km with higher power consumption than XBee. Lora communication can achieve long-range data transfer with lower power consumption than XBee, but its data rate is pretty low [8]. This low data rate can decrease the reliability of the wireless

communication that needs a high data rate. Thus, cost-effective and reliable wireless communication is required in data logging systems.

In addition to data logging, SCADA programs are often purchased to monitor system operation in industries. However, SCADA programs with high reliability and quality are not accessible for remote residences. Supervising remote systems requires long-distance transmitters and receivers, which increases the complexity and cost of the whole system. In comparison, free online platforms for Internet of Things (IoT), such as Thingspeak, Ericsson, and so on, offer free visualization and download functions for limited measurements from a system [8]. These platforms can be applied as low-cost monitors for remote systems. MATLAB can analyze the operation of these systems. Thus, the data visualization and analysis of the isolated PV system on an IoT platform and MATLAB, respectively, are required for the data logging system.

2. Overall System Setup

The whole system includes a PV system, weather data logger, PV system data logger, radio transmission and web transfer system on a PC. Its schematic is shown in Figure 1 below.

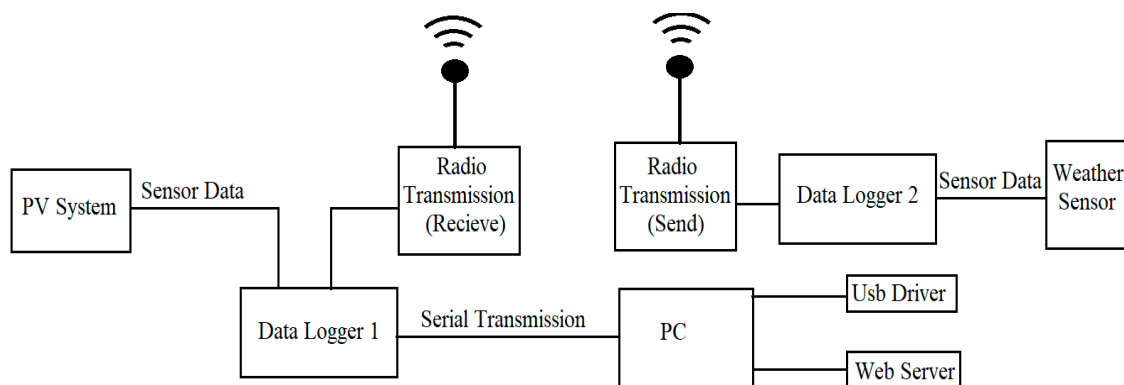


Figure 1. Overall system schematic.

Figure 1 shows the weather data which is first sent to data logger 2 and transmitted to data logger 1 through radio transmission. Then, by serial transmission data logger, 1 sends all the sensor data to a PC. The PC transfers the data to a local USB server and web server (Thingspeak). Since the real overall setup is too large to be included in one picture, the structure for the data logging system is shown in Figures 2 and 3, respectively. The complete system wiring diagram is shown in Figure 4.

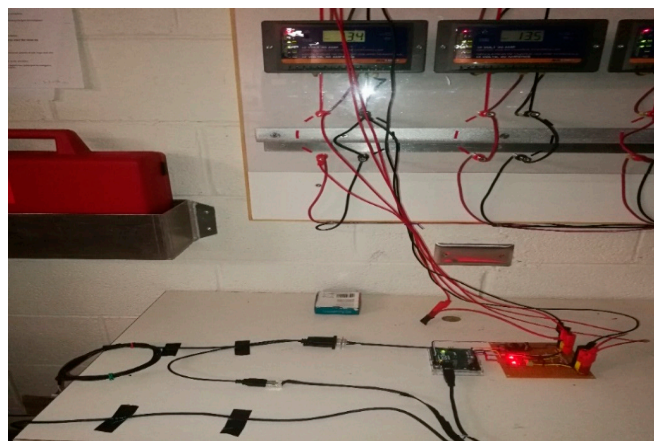


Figure 2. Lab setup of data logging system.

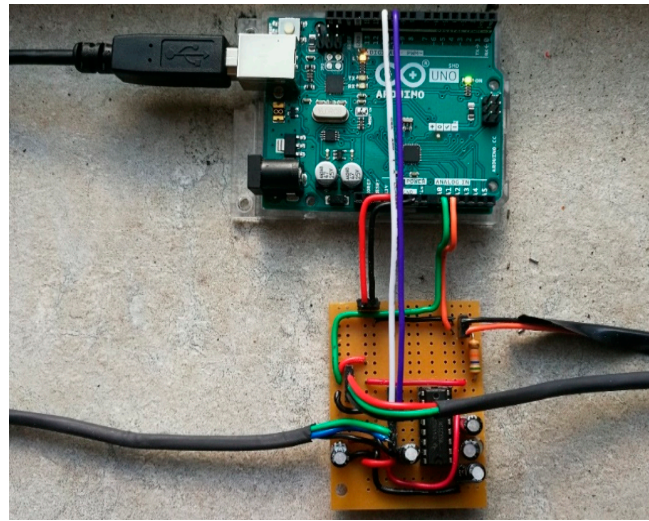


Figure 3. Weather data logger.

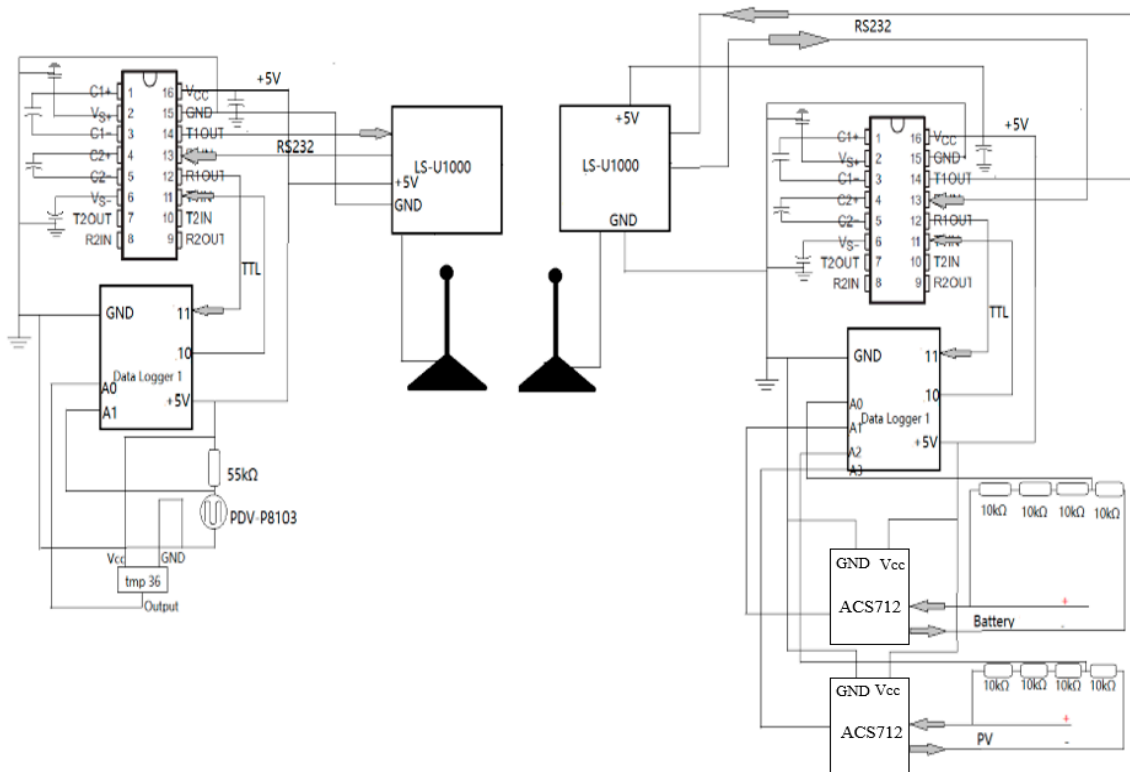


Figure 4. Detailed wiring diagram of data logging system.

3. PV System

The overall PV system consists of a PV array, maximum power point tracker (MPPT) and battery. The PV array is comprised of two PV panels (each with the 12 V output voltage and 130 W output power) in parallel [9]. The nominal voltage and current of MPPT are 12 V and 30 A respectively. Other detailed information is listed in reference [10]. The nominal voltage of the battery is 12 V, and the nominal capacity of the battery is 105 Ah [11]. The overall PV system is shown in Figure 5.

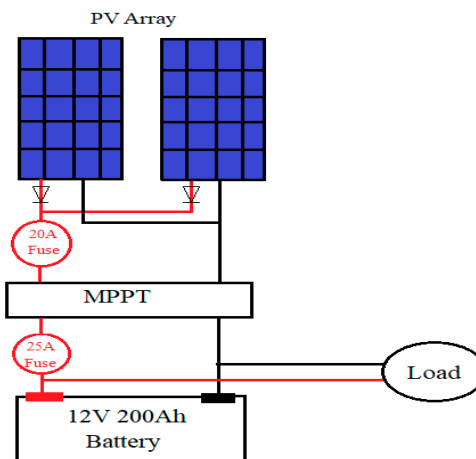


Figure 5. Overall photovoltaic (PV) system.

Figure 5 shows that during daytime, the PV array transforms the solar radiation to electricity through photovoltaic effect, then transmits the electrical power to MPPT. MPPT is a DC–DC converter which is used to reach the maximum power output of the PV array for the given temperature and solar radiation by adjusting the output voltage of the PV array. After the MPPT, the electricity charges the battery and supplies the load. During nights when there is no solar radiation, the output voltage and power of the PV array drops to zero. The blocking diodes at the output of the PV panels prevent the battery discharging during these periods. The PV system in the lab (without PV array) is shown in Figure 6 below.

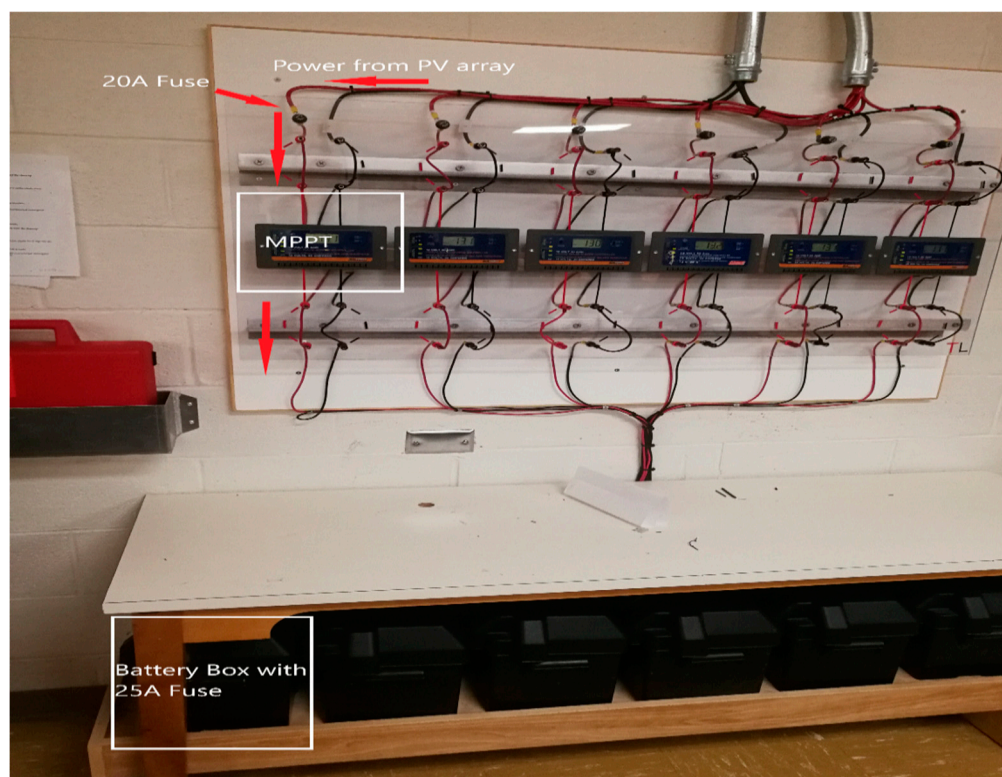


Figure 6. PV system in lab (without PV array).

4. Sensors

There are two sensor circuits, one for collecting the PV system's data and the other for the weather data. The sections below will discuss the sensors and sensor circuits in detail.

4.1. Sensors in PV System

The sensors in the PV system include two current sensors (same version) and two voltage dividers. The ratios (primary voltage to secondary voltage) of the voltage dividers of battery voltage and PV voltage are 4:1 and 6.56:1 respectively. The PV voltage divider and battery voltage divider are composed of four resistors, which are shown in Figure 7 below.

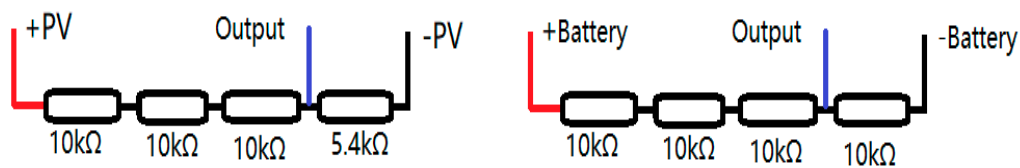


Figure 7. Voltage divider.

The outputs in the Figure above are analog values. Arduino will transform the analog values into digital values by Equation (1).

$$Data = measured_value \times 1024 / reference_value \quad (1)$$

Thus, the real-time measured voltage should be calculated by Equation (2).

$$V_{real} = Data \times \frac{Ratio}{204.6} (V) \quad (2)$$

The ratio for the PV voltage divider is 6.56, and 4 for the battery voltage divider. The adopted current sensor is ACS712 [12], which is shown in Figure 8 below.



Figure 8. Current sensor.

The offset output voltage of the current sensor is 2.5 V, its current range is from -20 A to 20 A and its sensitivity is 180 mV/A. Other detailed data of the current sensor is shown in [12]. The testing results of the current sensor measuring PV output current and the current sensor measuring the battery input current are shown in Tables 1 and 2, respectively:

Table 1. Test result of photovoltaic (PV) current sensor.

Input Current (A)	Output Voltage (V)	Average Sensitivity (V/A)
0	2.487	0.187
1	2.671	
2	2.855	
2.38	2.928	

Table 2. Testing result of battery current sensor.

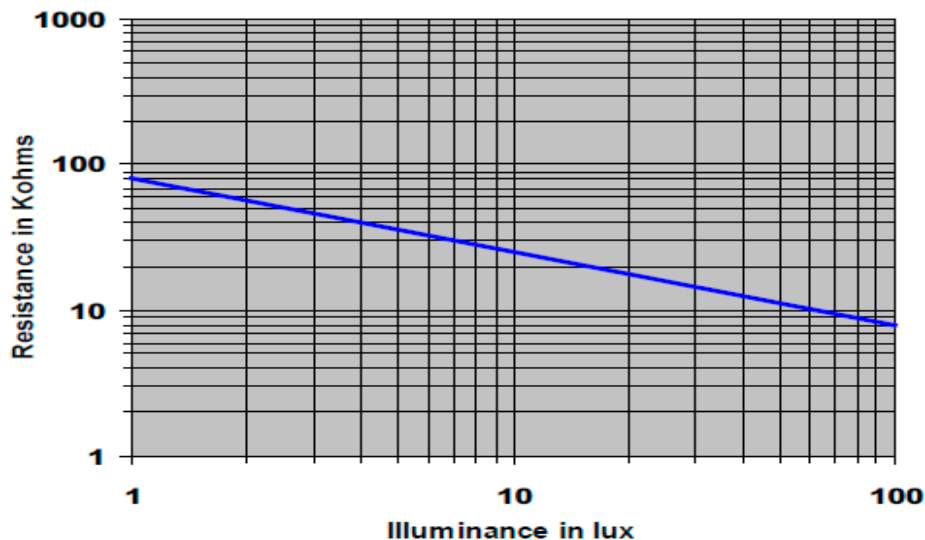
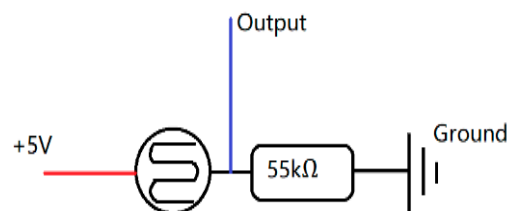
Input Current (A)	Output Voltage (V)	Average Sensitivity (V/A)
0	2.499	0.1872
1	2.683	
1.99	2.867	
2.46	2.958	

Similarly, the real-time value of the current should be calculated by Equation (3) below.

$$I_{real} = \frac{Data}{38.51} - 13.32 \text{ (A)} \quad (3)$$

4.2. Sensors for Weather Data

Collected weather data includes the data of solar radiation and temperature. The light dependent resistor is PDV-P8103. Its sensitivity ($\lg(R100) - \lg(R10)/\lg(E100) - \lg(E10)$) is 0.75, and its typical performance is shown in Figure 9. Other detailed data are shown in reference [13]. The complete circuit of the solar sensor is shown in Figure 10.

**Figure 9.** Typical performance of PDV-P8103.**Figure 10.** Circuit of the solar sensor.

According to the given data and the performance curve of the light dependent resistor, the solar radiation is calculated by Equation (4) below.

$$Ra = 0.0079 \times 10^{(2.5311 - \lg^{(55.7 \times data / (1170.31 - data))})} \quad (4)$$

The adopted temperature sensor is tmp36. Its offset output voltage is 0.5 V, its output voltage scaling is 10 mV/°C, and its output voltage at 25 °C is 750 mV. Other detailed data are in Reference [14]. The real-time temperature is calculated by Equation (5) below.

$$T = \text{Data}/1.945 - 50 \quad (5)$$

5. Radio Communication System

The radio communication system is used to transmit data between data logger 1 and data logger 2. The radio transmission system is built based on 2 LS-U1000 RF modules with RS232 interface. Either LS-U1000 RF module can be the transceiver or receiver without programming. The RS232-TTL interface boosts the digital signals in the TTL level (0–5 V) from one data logger to RS232 level (± 8.5 V). LS-U1000 RF circuit modulates the boosted signal to the electric current and sends it to the transceiver antenna, which transforms the electric current into radio waves. The receiving antenna (conductor) then converts the radio wave into electric wave (current). Then, the receiver (LS-U1000) transforms the current into digital signals, and sends it to another data logger through RS232-TTL interface. The overall communication system is shown in Figure 11.

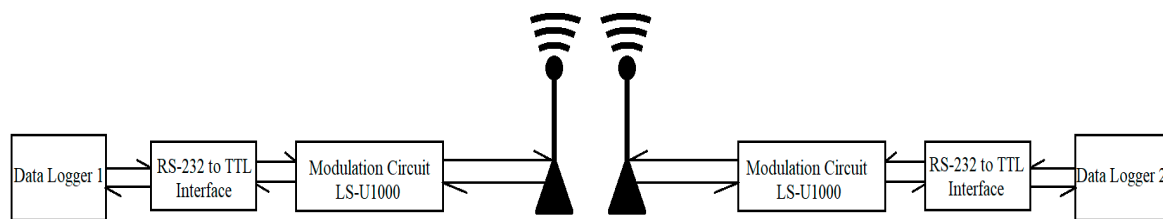


Figure 11. Overall radio communication system.

Since LS-U1000 communicates with other devices by RS232 protocol, the RS232-TTL interface circuit is needed to boost the TTL-level (0–5 V) signals from the data loggers up to RS232 level (± 8.5 V), or level down the RS232-level signals to TTL level to the data loggers. Otherwise, the signals in the RS232 level may destroy the data loggers. The detailed design of the RS232-TTL interface circuit is described in Section 5 below.

RS232-TTL Interface

MAX232 is the foundation of the RS232-TTL interface. The interface uses a capacitive voltage generator to supply RS232 voltage level [5]. It has two receivers and two transceivers. Each receiver levels down the RS232 level signals to TTL level and each driver boosts up the TTL level signals to RS232 level. Other detailed data are shown in reference [15]. Since the interface circuits for transmitter and receiver are equally similar, the only one-side interface circuit is shown in Figure 12 below.

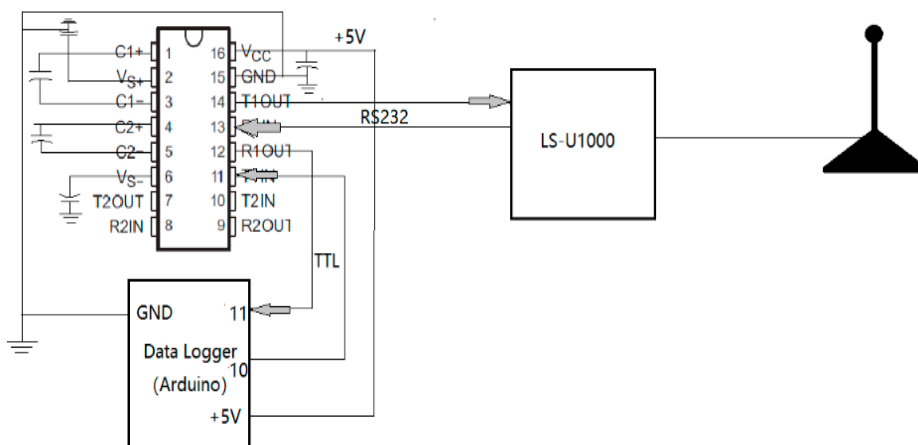


Figure 12. Detailed RS232-TTL interface.

Figure 12 shows that the output signal (TTL level) from pin 11 of one Arduino board is transmitted to the input of MAX232's driver #1. The output signals from driver #1 are then boosted up to RS232 level and sent to LS-U1000, which transforms the signal to electric current and sends it to the antenna. The antenna converts the electric wave into a radio wave, which is then received by receivers and converted into TTL-level signals to the other Arduino logger.

6. Programs in Arduino Boards and Python Script

6.1. Program of Arduino Boards (Data Loggers)

Arduino boards collect sensor data, calibrate it and send it to the PC for further processing. Before this process, data logger 2 has to receive the sensor data collected by data logger 1. Synchronization is thus needed between these two data loggers, to obtain the accurate real-time measurement of solar radiation and temperature from data logger 1. The flow charts of the program of these two data loggers are shown in Figures 13 and 14 below.

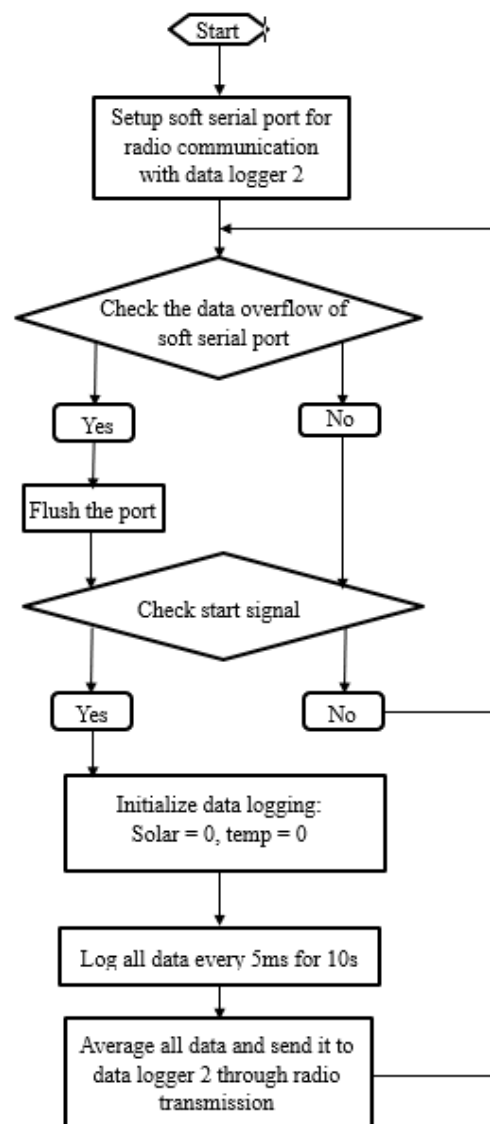


Figure 13. Program flow chart of data logger 1.

Figure 14 shows that data logger 2 not only collects the PV system's data (current and voltage) but also receives the weather data from data logger 1 (Figure 13) through radio transmission. After

setting up the serial and software serial port that transmits data to the PC and data logger 1, data logger 2 flushes the serial buffer to send the most recent data to the PC. After flushing the serial port connected with the PC, it checks the overflow of the software serial buffer, which guarantees that data logger 2 receives the most recent data from logger 1. Then, data logger 2 sends the start signal 's' to data logger 1, and then both loggers begin collecting sensor data. This process includes initializing variables and collecting and calibrating data. After processing the sensor data, data logger 2 waits for the data from data logger 1. Once the data logger 2 senses the data from data logger 1, it will send all the sensor data to the PC and start the next loop after 20 s; otherwise, it will directly start the loop after sleeping 20 s.

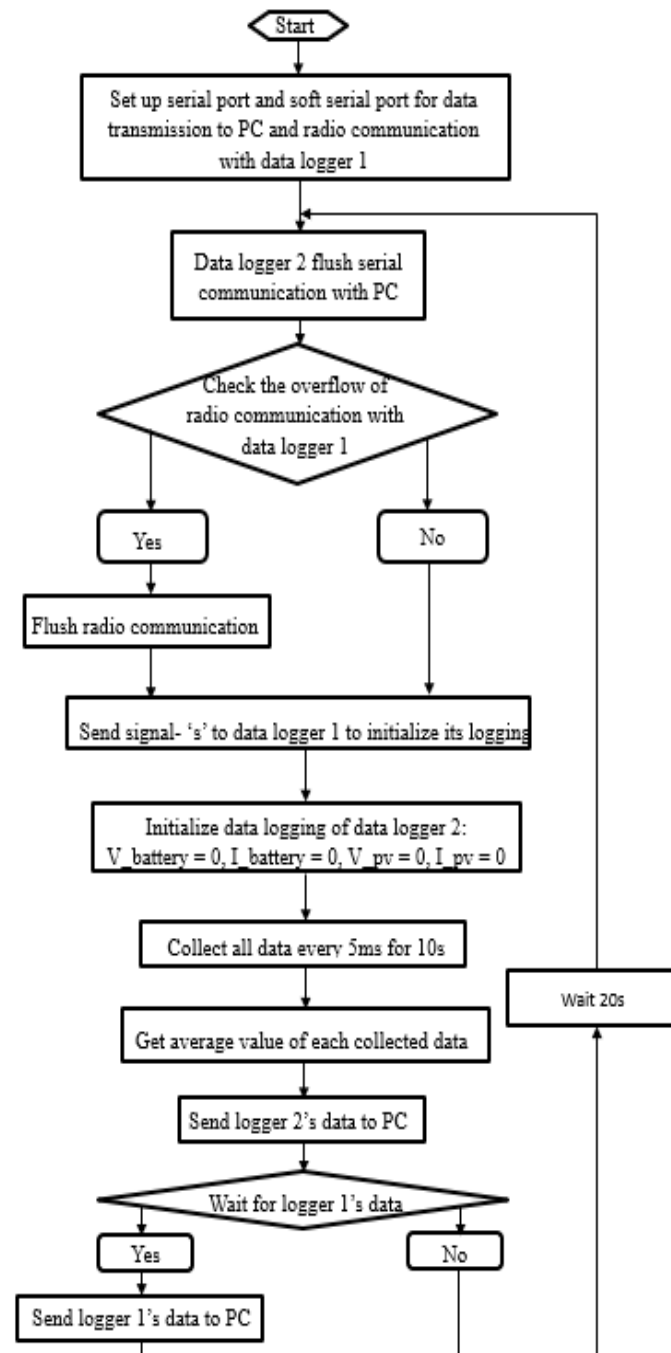


Figure 14. Program flow charts of data logger 2.

6.2. Python Program on PC

The python program receives and processes the data from logger 2 back to real-time measurements, then transmits them to the Thingspeak server and a local hard drive. However, for long-time data transfer, the structure of the program should be strengthened to prevent it from being stopped by small errors. The program firstly initializes the serial communication with data logger 2, the URL transmission with the Thingspeak server and data storage on the local USB drive. Then, it checks the dataset from data logger 2 at the serial port. If it exists, the program decodes it to string from binary mode; otherwise, it loops back to check the dataset. If there is no error during decoding, the program extracts each data (solar radiation, temperature, battery's current and voltage, and PV array's current and voltage) from the dataset; otherwise, the dataset becomes empty, and the subsequent data will also be empty. The program then checks the availability of each extracted data (no matter if it is empty or not) and transforms unavailable data to empty data. Then, the data (whether it is empty or not) is transformed from string to float data and processed to real-time measurements according to Equations (2)–(5). Next, the program checks the availability of URL transmission, posts the real-time data to the Thingspeak server if the transmission is available, and then transforms the data to the USB storage; otherwise the program sleeps 5 s before the next step, in which the program saves the sensed data and measurement time to a local USB drive. Then, another cycle of the program begins after 5 s' sleep and flushes the serial buffer again. The detailed flow chart of the program is shown in Figure 15 below.

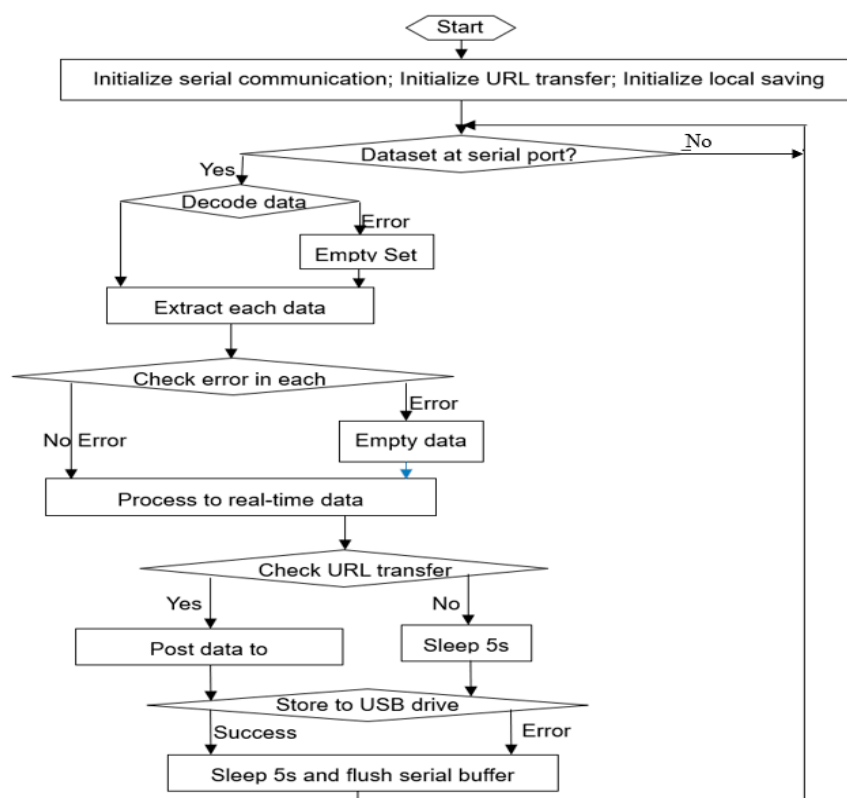


Figure 15. Flow chart of python program.

7. Data Visualization and Analysis

7.1. Basic Settings on Thingspeak Server

Thingspeak is an open online platform to collect, analysis and act on sensor data [16]. To fully visualize all the collected data, six fields are set up. Field 1 is set up for the battery voltage, field 2 for battery current, field 3 for PV voltage, field 4 for PV current, field 5 for solar radiation and field 6 for

temperature. The message of connection failure is set to send to users' Twitter account to alarm the users (shown in Figure 16).



Figure 16. Connection failure message on the user's Twitter account.

7.2. Data Visualization

The data are collected every 30 s for 16 days. The long-time testing proves that the overall system can achieve long-time robust data log and transfer. One day's data on Thingspeak are shown in Figures 17–19.

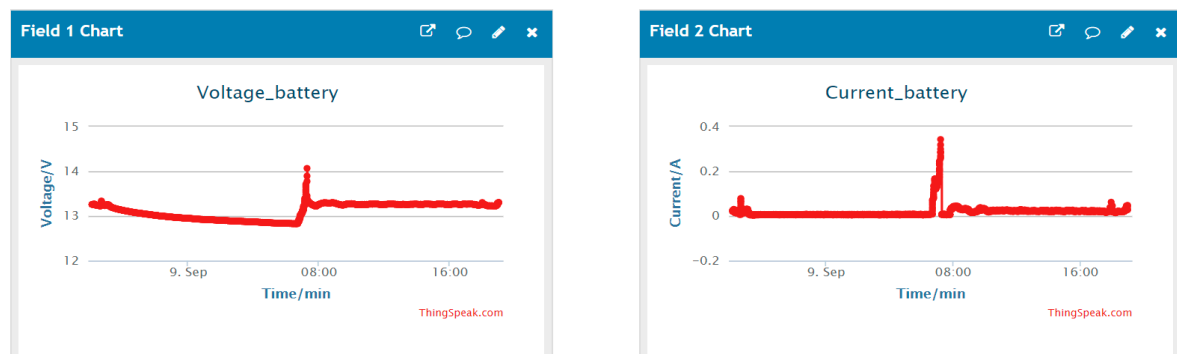


Figure 17. Voltage and current of battery in one day.

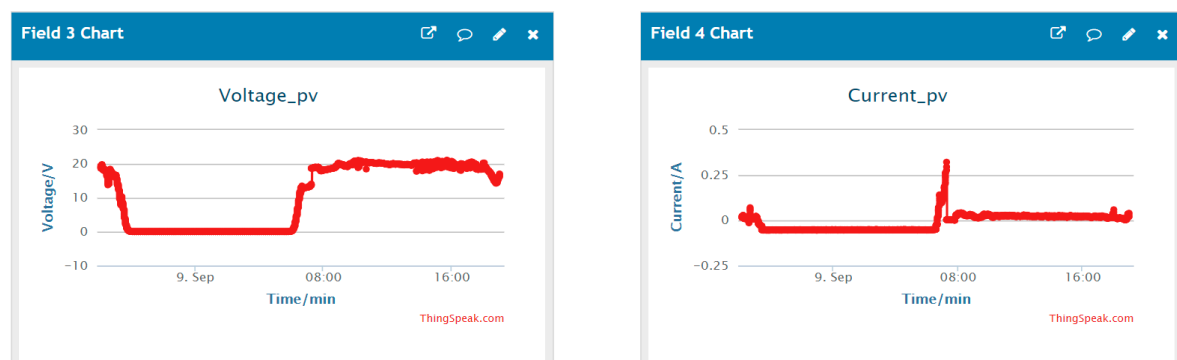


Figure 18. Voltage and current of PV array in one day.

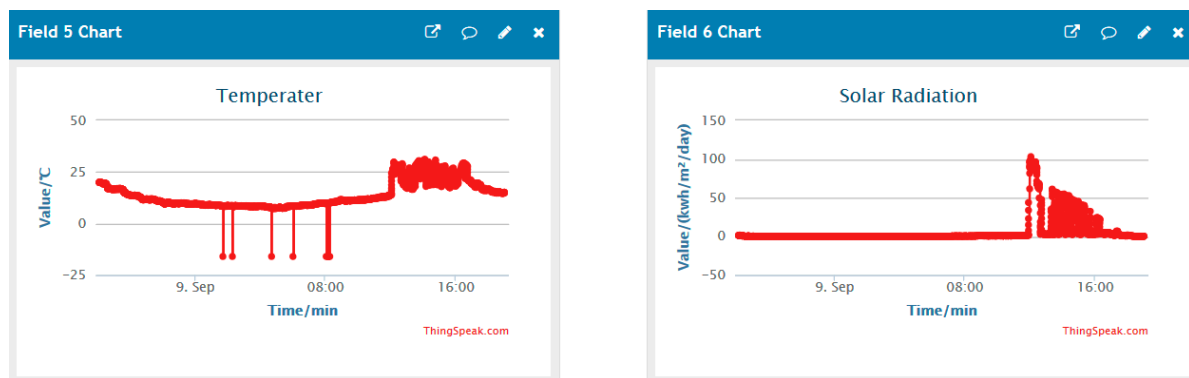


Figure 19. Temperature and solar radiation in one day.

According to the comparison among the solar radiation, current and voltage data of the PV system, solar radiation is proportional to the values of input-output voltage and current of MPPT. During daytime, the output voltage of MPPT (battery voltage) is lower than the input voltage of MPPT. There are two reasons for this phenomenon. The first is that the PV array's output voltage at the maximum power point is always smaller when the voltage of the battery is low. The second reason is that MPPT's DC/DC converter is open-circuit for the most time of each cycle due to the near fully charged battery. Thus, MPPT's output voltage is closer to the battery voltage. Figure 18 shows that the temperature is also proportional to solar radiation. Please note that the solar radiation is not equal to the radiation received by the PV array. That is caused by the shadow from the infrastructure near the light resistor (shown in Figure 20).

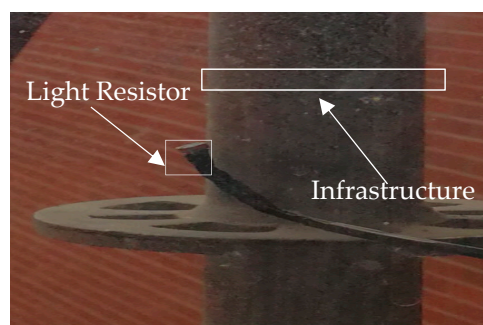


Figure 20. Obstruction of light resistor.

All collected sensor data are shown in Figure 21 below.

Figure 21 shows that the voltage and current of the battery and PV array and the temperature are proportional to solar radiation. The PV array begins to receive solar radiation and charge the battery around 6:00 a.m. every day. Simultaneously, the output voltage of PV array resumes to 20 V from 0 V, and its output current begins to increase to a positive value. Due to one night's discharge, the capacity of the battery drops. Moreover, MPPT starts its functions. During 5:00 a.m. to 8:00 a.m., MPPT increases its output voltage (battery voltage) to track PV array's maximum output power, which also increases the battery current and PV current until the battery is fully charged. The current and voltage of the system, then, keeps constant for the rest of the daytime. Around 8:00 p.m. solar radiation declines to a specific value when the output voltage and current of the PV array begins to decrease. At the same time, the power from the PV array is not enough to compensate for the self-discharge energy of the battery, so the voltage and current of the battery also decrease. During the night without solar radiation, the output voltage of the PV array becomes zero, so do the output current of the PV array and MPPT. The output voltage of MPPT is equal to the battery voltage during the same period since the blocking diode of the PV array and the power electronic device of MPPT block the

electrical connection between the PV array and battery. Both Figures 18 and 21 show that there are some interruptions in the temperature measurement on the last day. This fault is caused by an unreliable temperature-measurement circuit, which needs to be improved in future research.

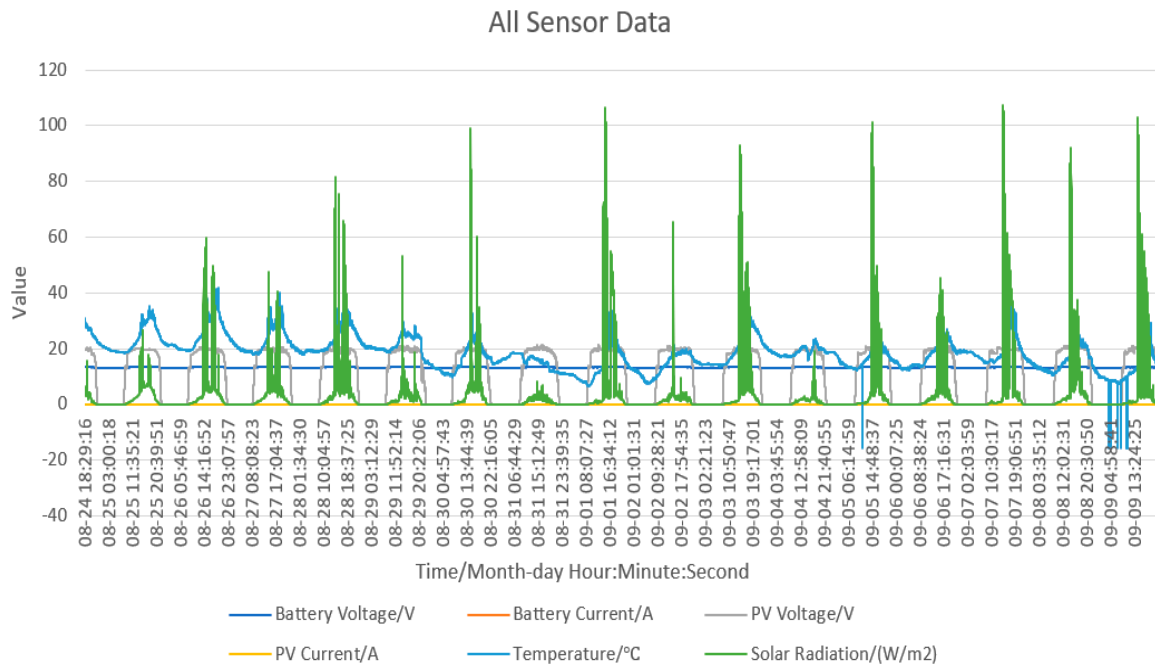


Figure 21. All sensor data.

7.3. Data Analysis

The battery is near fully charged all the time, and the measurement of solar radiation is frequently disturbed by the shadow of the infrastructure mentioned in Section 6.2. Thus, it is not reliable to check the operation of the PV array based on the collected data. This section only describes the efficiency and daily energy supply of the MPPT in MATLAB based on the data collected on the Thingspeak server. The flow chart of data-analysis program in MATLAB is shown in Figure 22 below.

Figure 22 shows that the efficiency and output power of MPPT are initialized as two $1 \times 44,014$ arrays. Each element of arrays result from the process of each loop. When there is no solar radiation, the output current, and input current of MPPT sometimes are less than zero due to battery discharge, and the MPPT stops functioning during this time. Thus, in each loop, if any current is less than zero, the efficiency is set to zero. Because of the measurement error of current and voltage, the efficiency is sometimes equal to or larger than 1. The maximum efficiency of the MPPT is then set to 0.98. After efficiency calculation, the program calculates the daily energy supply from the 1st to 15th day every 2880 loops, and the daily energy of the 16th day is calculated by subtracting the supplied energy in the previous 15 days from the total energy provided after the loop. The output power and efficiency of the MPPT are shown in Figure 23.

Figure 23 shows MPPT's efficiency during the morning and evening which is the largest during each day. After discharging for one night, the voltage and capacity of the battery are the lowest of the day. The solar radiation during this time is also small, MPPT operates in high efficiency to gain maximum power from the PV array to charge the battery. Similarly, during the evening when solar radiation becomes a shallow level, MPPT operates in high efficiency to extract maximum power from the PV array to compensate for the self-discharge power of the battery. Due to higher solar radiation during the rest of the daytime, MPPT's efficiency is relatively lower. Therefore, the power output of the PV array is limited, which protects the battery from overcharging. The energy charged to the battery each day is shown in Figure 24 below.

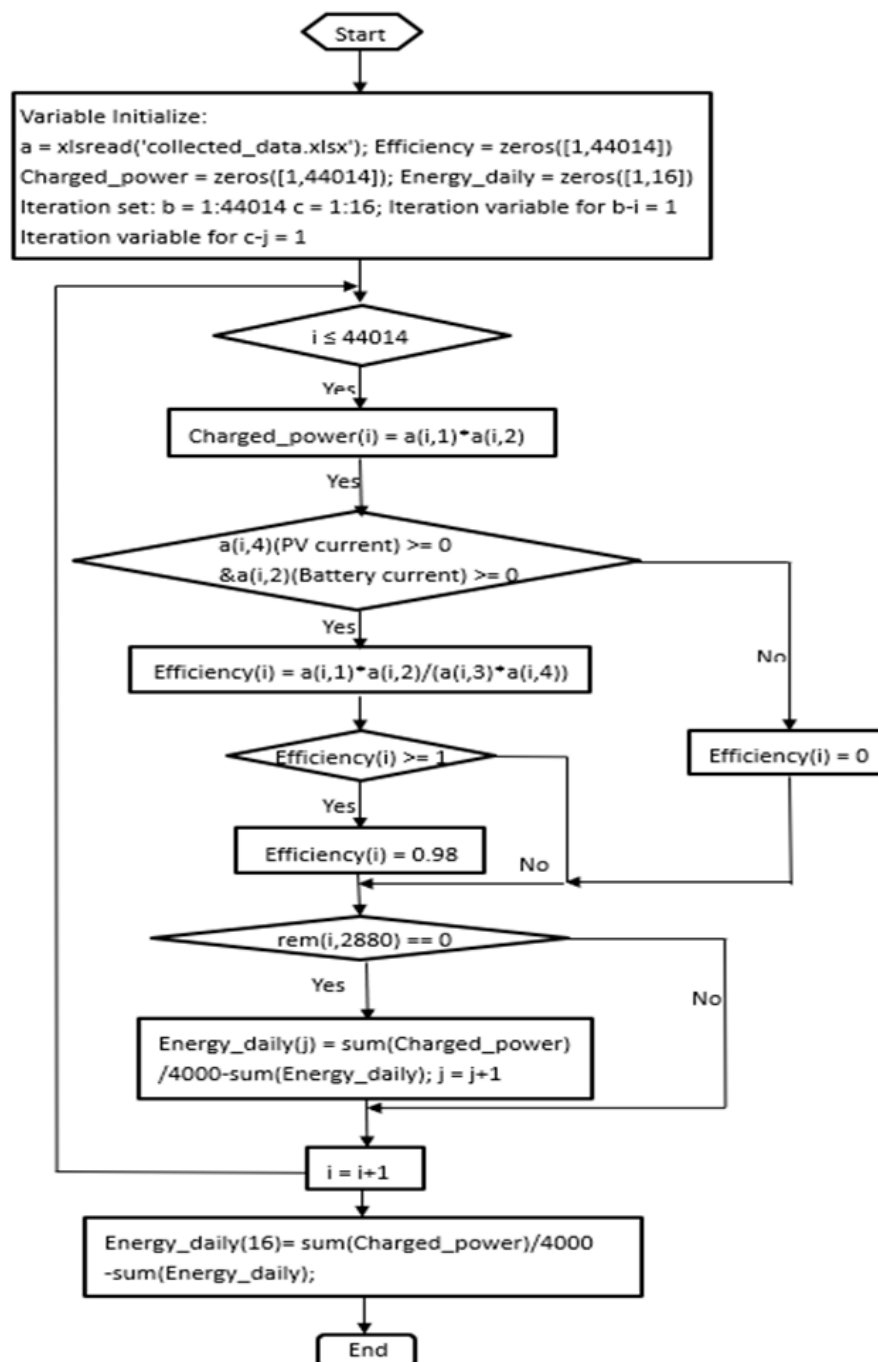


Figure 22. Flow chart of data analysis in MATLAB.

Figure 24 shows that the energy supplied to the battery is very low on the 16th day since the lab manager stopped the system in the middle of the 16th day. Except for that, the generated energy in other days is very even due to the stable weather over these 16 days.

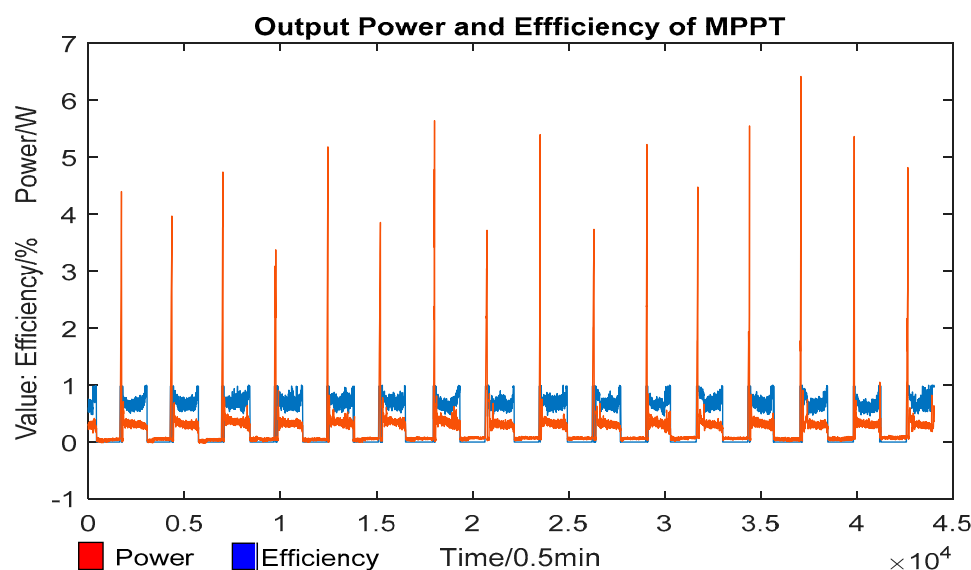


Figure 23. Output power and efficiency of maximum power point tracker (MPPT).

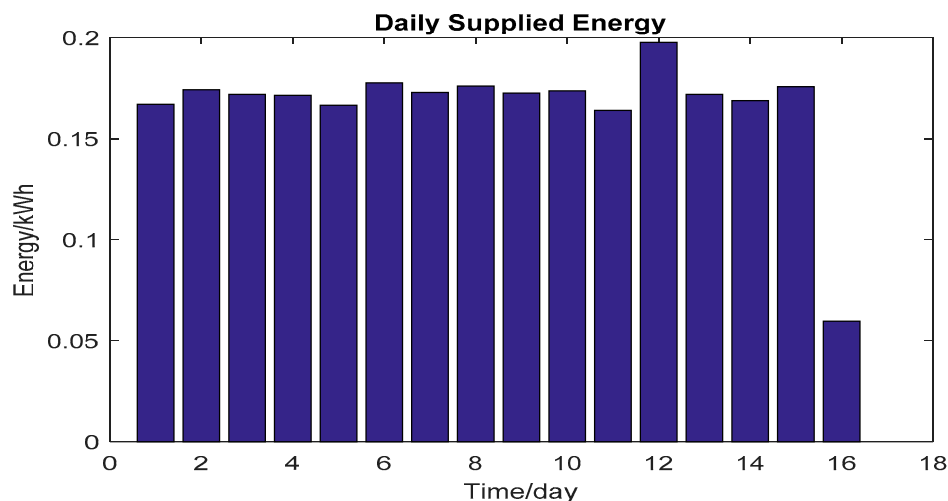


Figure 24. Daily supplied energy.

8. Conclusions

This paper has presented the design of data logger, data storage and data visualization system, all circuit details, all logged data and fundamental data analysis. The designed system performed well during the test period of 16 days. Thus, this open-source and low-cost data logger can replace commercial data loggers and monitor the operation of isolated PV systems. Issues faced during data logging and design are also described in this paper, such as the solar radiation block from nearby infrastructure, uncalibrated solar radiation, and temperature sensor circuits, unexpected interruption and so on. More detailed data analysis of the PV array's operation will be presented in future research after fixing the issues discussed above.

Author Contributions: B.J. did most of the research. M.T.I. supervised him during the research, reviewed and corrected the manuscript, provided all required components and research ideas.

Acknowledgments: This research was funded by Gansu Liujiaxia Hydropower Co., Ltd. The authors would like to acknowledge all the support and help from friends, family, and Memorial University of Newfoundland.

Conflicts of Interest: The authors of this project announce that all the devices and software used in this work are selected on a professional basis. Furthermore, the authors confirm that there are no risks that lead to direct or potential conflicts of interest.

References

1. Sánchez-Pacheco, F.J.; Sotorriño-Ruiz, P.J.; Heredia-Larrubia, J.R.; Pérez-Hidalgo, F.; de Cardona, M.S. PLC-Based PV Plants Smart Monitoring System: Field Measurements and Uncertainty Estimation. *IEEE Trans. Instrum. Meas.* **2014**, *63*, 2215–2222. [CrossRef]
2. Rezk, H.; Tyukhov, I.; Al-Dhaifallah, M.; Tikhonov, A. Performance of Data Acquisition System for Monitoring PV System Parameters. *Measurement* **2017**, *104*, 204–211. [CrossRef]
3. Munandar, D.; Syamsi, D. Data logger Management Software Design for Maintenance and Utility in Remote. In Proceedings of the 1st International Conference on Information Technology, Computer and Electrical Engineering, Semarang, Indonesia, 8 November 2014.
4. Ramdan, A.; Astrid, M. Design and Implementation of Data Storage System Using USB Flash Drive in A Microcontroller Based Data Logger. In Proceedings of the International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), Bandung, Indonesia, 29–30 October 2015.
5. Hadiatna, F.; Hindersah, H.; Yolanda, D.; Triawan, M.A. Design and Implementation of Data Logger Using Lossless Data Compression Method for Internet of Things. In Proceedings of the 2016 IEEE 6th International Conference on System Engineering and Technology (ICSET), Bandung, Indonesia, 3–4 October 2016.
6. Touati, F.; Al-Hitmi, M.A.; Chowdhury, N.A.; Hamad, J.A.; San Pedro Gonzales, A.J.R. Investigation of solar PV performance under Doha weather using a customized measurement and monitoring system. *Renew. Energ.* **2016**, *89*, 564–577. [CrossRef]
7. Available online: <https://en.m.wikipedia.org/wiki/Xbee> (accessed on 4 April 2019).
8. Terashmila Lasagani, K.A.; Iqbal, T.; Mann, G. Data Logging and Control of a Remote Inverter Using Lora and Power Line Communication. *EPE J.* **2018**, *10*, 351–365. [CrossRef]
9. Available online: <https://www.topsolarpanel.com/product/sunforce-37130-130w-crystalline-solar-panel> (accessed on 4 April 2019).
10. Coleman. *Coleman 12V 30Amp Digital Charge Controller User's Manual*; Coleman: Wichita, KS, USA.
11. Available online: <https://www.canadiantire.ca/en/pdp/motomaster-nautilus-group-31-deep-cycle-battery-0103199p.html> (accessed on 4 April 2019).
12. Allegro MicroSystems, LLC. *Fully Integrated, Hall Effect-Based Linear Current Sensor IC with 2.1kV RMS Isolation and a Low-Resistance Current Conductor*; Allegro MicroSystems, LLC: Worcester, MA, USA, 2017.
13. Luna Optoelectronics. *CDS Photoconductive Photocells PDV-P8103*; Luna Optoelectronics: Camarillo, CA, USA, 2016.
14. Analog Devices. *Low Voltage Temperature Sensors: TMP35/TMP36/TMP37*; Analog Devices: Norwood, MA, USA.
15. Texas Instruments. *MAX232x Dual EIA-262 Drivers/Receivers*, 2nd ed.; Texas Instruments: Dallas, TX, USA, 2014.
16. Available online: <https://www.mathworks.com/help/thingspeak/> (accessed on 4 April 2019).



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).